A Distributed Approach For Multi-Constrained Path Selection And Routing Optimization[1]

Zhenjiang Li[†]
zhjli@soe.ucsc.edu
[†]Computer Engineering, Univ. of California, Santa Cruz
1156 high street, Santa Cruz, CA 95064, USA
Phone:1-831-4595436, Fax: 1-831-4594829

J.J. Garcia-Luna-Aceves [†*]
jj@soe.ucsc.edu
[*]Palo Alto Research Center (PARC)
3333 Coyote Hill Road
Palo Alto, CA 94304, USA

Abstract

Multi-constrained path (MCP) selection, in which the key objective is to search for feasible paths satisfying multiple routing constraints simultaneously, is known to be an NP-Complete problem. Multi-constrained path optimization (MCPO) is different from MCP mainly in that, the feasible paths selected should also be optimal with regard to an optimization metric, which makes path computation in MCPO even harder.

We propose a fully distributed multi-constrained path optimization routing (MPOR) protocol that solves the general k-constrained path selection and routing optimization problems. MPOR computes paths using distance vectors exchanged only amongst neighboring nodes and does not require the maintenance of global network state about the topology or resources; supports hop-by-hop, connectionless routing of data packets, and implements constrained path optimization by distributively constructing an x-optimal path set (i.e., the shortest, the second shortest and up to the xth shortest path in terms of the optimization metric) for each destination at each node. Simulations show that MPOR has satisfactory routing success ratios for multi-constrained path selection, and performs consistently with varying number of constraints. For constrained path optimization, MPOR has high probabilities of finding feasible paths that are also optimal or near-optimal for the given optimization metric.

[1]This work was supported in part by the Baskin Chair of Computer Engineering at UCSC, the National Science Foundation under Grant CNS-0435522, the U.S. Army Research Office under grant No. W911NF-05-1-0246. Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the funding agencies.

1 Introduction

A central problem in QoS routing consists of finding *feasible paths* that satisfy multiple performance oriented constraints (e.g., bandwidth, end-to-end delay and jitter) between a source and a destination. In some scenarios, these feasible paths must also be optimized with regard to (w.r.t.) a given optimization metric (e.g., hop-constrained widest-path routing, delay-constrained least-cost path routing). The first problem is known as the NP-hard multi-constrained path (MCP) selection problem, and the second is called the multi-constrained path optimization (MCPO) problem. Many heuristic algorithms have been proposed to solve each problem. However, only a few approaches address both problems simultaneously, especially for the general k-constrained path optimization problem.

To date, most existing constrained routing algorithms deal with routing subject to two constraints, or its optimization case–the *restricted shortest path (RSP)* problem, in which the task is to seek paths that satisfy one constraint while optimizing another metric. Such MCP and RSP problems are not strong NP-complete, in that there are pseudo-polynomial running time algorithms to solve them exactly, and the computational complexity depends on the values of link weight in addition to the network size [5]. However, their complexity is prohibitively high when the values of link weight become large.

Based on the latter observation above, Chen and Nahrstedt [1] proposed to scale one component of the link weight down to an integer that is less than $\lceil \frac{w_i \cdot x}{c_i} \rceil$, where x is a pre-defined integer and c_i is the corresponding constraint on weight component w_i. They prove that the problem after weight scaling is polynomially solvable by an extended version of Dijkstra's (or Bellman-Ford) algorithm, and any solution to the latter is also a solution to the original MCP problem. The running time is $O(x^2N^2)$ when the extended Dijkstra's algorithm is used; and it is $O(xEN)$ when the

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
A Distributed Approach for Multi-Constrained Path Selection and Routing Optimization	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California Santa Cruz,Computer Engineering,1156 High Street,Santa Cruz,CA,95064**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**9**	

extended Bellman-Ford algorithm is used, where E and N are the number of links and nodes, respectively. However, as shown in [7], satisfactory performance is achievable only when parameter x is large enough (e.g., $x = 10$), which incurs considerable computation complexity.

Another commonly used scheme for MCP is to define a good link-cost (or path-weight) aggregation function based on the routing metrics and the given constraints. Then any shortest path algorithm can be used to compute the shortest path w.r.t. the single aggregated metric. Jaffe [5] was the first to use a linear link-cost function $w(u,v) = \alpha w_1(u,v) + \beta w_2(u,v)$, in which $\alpha, \beta \in Z^+$. Ever since, both linear and non-linear aggregation functions have been proposed. The major limitation of this approach is that the ability to find feasible paths based on an aggregated metric largely depends on the *quality* of the functions being used, and most of them are empirical heuristics. Consequently, the shortest path computed w.r.t. the single aggregated metric may not simultaneously satisfy the multiple constraints being considered.

The general k-constrained path computation has received far less attention than the 2-constrained MCP or RSP. Yuan [14] generalized the ideas of link-weight scaling as the limited-granularity (LG) heuristic, and also proposed the limited-path (LP) heuristic in which x non-dominated paths are maintained at each node. Then an extended Bellman-Ford algorithm is used to work with one of them to solve the general k-constrained MCP problem. The running time of Yuan's algorithm is $|X| \cdot NE$, where $|X|$ is the size of the table to use, for LG; and it is x^2NE, where x is the number of path to maintain, for LP. The performance and complexity of Yuan's heuristics depend on the number of possible values to which link weight can be scaled down, or the number of paths to maintain at each node. For Yuan's algorithms to work correctly, global network state has to be available to nodes that perform the path computation.

The implementation strategies for QoS routing can be classified into centralized source routing and distributed routing. Most constrained routing algorithms proposed to date use centralized approaches, i.e., they compute feasible paths at each source, and simply assume the availability of global network state whenever they are needed. Centralized MCP algorithms suffer from high computation complexity at the source nodes, sluggish response to network changes, and excessive overhead caused by disseminating topology and resource information throughout the network, which significantly limit their scalability. Due to the use of source routing, a complete feasible path towards the destination usually needs to be established before actual data are sent out (i.e., connection oriented), or appended to the header of every data packet, which may increase the packet delivery delay and consume large fraction of the bandwidth when the paths being used are long.

Distributed MCP algorithms compute feasible paths locally at each node, and forward packets based only on their destination addresses on a hop-by-hop basis. Distributed routing is more responsive, robust and scalable than centralized schemes mainly in that, nodes independently make routing decision and therefore are able to respond to network dynamics quickly.

Mieghem et al proposed a hop-by-hop destination based only (HbHDBO) QoS routing algorithm [9] Based on TAMCRA [10] and its exact modification SAMCRA. HbHDBO has the worst case complexity $O(kNlog(kN) + k^2CE)$, where k is the number of non-dominated paths maintained at each node and C is the number of constraints being concerned with. A node that runs HbHDBO uses a modified Dijkstra's algorithm to compute k non-dominated paths for each destination, using a non-linear weight function $w(p) = max\left(\frac{w_i(p)}{c_i}\right)$, where c_i is the constraint on metric w_i. For HbHDBO to work correctly, global network state is required at each node, and routing constraints must also be known a priori. However, in the worst case, k can grow exponentially large; and the performance of HbHDBO also varies with the number of routing constraints.

Sobrinho uses an algebraic approach and investigates the path optimization problem in the context of Hop-by-Hop QoS routing[12]. In Sobrinho's algebraic framework, routing is separated into path weight functions that define routing optimization requirements (e.g., widest-shortest path, most-reliable path), and the algorithms that compute the optimal paths based on the aggregated metric defined by the weight function being used. Sobrinho establishes the algebraic properties that a path weight function must have, in order for any routing algorithm to converge correctly (i.e., to give the optimal paths w.r.t. the aggregated metric). Though the results obtained by Sobrinho establish a generalized framework for QoS-oriented path optimization, they cannot be simply applied to constrained path optimization, because paths are optimized only with respect to the path weight function, rather than being computed to satisfy multiple constraints.

It is clear from the above summary of related work that existing distributed MCP algorithms require a consistent view of global network state at each node, and some even assume that the distribution of arriving routing constraints is known, which is not true or hardly achievable in practice. On the other hand, most proposed QoS path optimization algorithms optimize routing based only on an aggregated metric, and therefore cannot be applied to path optimization subject to multiple constraints.

In this paper, we propose, analyze, and validate the multi-constrained path optimization routing (MPOR) protocol. Nodes that run MPOR use distance vectors to exchange route information only amongst neighbors, and do not require knowing the global network state or the potential ar-

riving constraints. MPOR solves the general k-constrained path optimization by distributively computing an x-optimal path set (the first x shortest paths w.r.t. the given optimization metric) for each destination in the network, in such a way that feasible paths can be found with a small value of x. Because multiple paths are maintained at each node, as in TAMCRA [10] and HbHDBO [9], MPOR also resolves the multi-constrained path (MCP) selection simultaneously. MPOR operates in line with IP routing (i.e., table-driven, hop-by-hop, and connection-less), and is able to optimize routing w.r.t. a singular metric (the case in the conventional MCPO), as well as any logical distance that considers multiple routing metrics, provided that the path function being used is both $monotone$ and $isotone$[1].

The rest of this paper is organized as follows. Section 2 introduces the network model and background knowledge used in our discussion. Section 3 describes the operation principles of MPOR, the ordered loop-free condition *o-LFC*, the process of path weight propagation, and the convergence property of MPOR. Section 4 presents the complexity analysis and Section 5 presents extensive simulation results of how effectively MPOR solves both multi-constrained path selection and routing optimization. Section 6 presents the concluding remarks.

2 System model

We model the network as a directed graph $G = \{V, L\}$, where V is the set of nodes and L is the set of links interconnecting the nodes. N and E are the cardinalities of V and L, i.e., $N = |V|$ and $E = |L|$, respectively.

We assume that each link $\{l_{u,v} | u, v \in V\}$ is associated with a link weight vector $w(u,v) = \{w_1, w_2 ... w_k\}$, in which w_i is an individual weight component, i.e., a single routing metric. Accordingly, any path p from a source to a destination can be assigned a path weight vector $w^p = \{w_1^p, w_2^p ... w_k^p\}$, where $w_i^p = \sum_{l_{u,v} \in p} w(u,v)$, if w_i is an additive metric (e.g., delay); or $w_i^p = min(w_i(u,v)),\ l_{u,v} \in p$, if w_i is a minimal metric (e.g., bandwidth) [2].

The *logical distance (LD)* of path p is given by a path function f^p based on the weights of its consisting links. In QoS routing, f^p is usually used to specify how the routing should be optimized. In the bandwidth-inversion shortest-path, e.g., $f^p = \sum_{l_{u,v}} \frac{1}{B(u,v)}, l_{u,v} \in p$, where $B(u,v)$ is the available bandwidth of $l_{u,v}$, such that the shortest path, in term of inverted bandwidth, is preferred; while in the most-reliable path routing, we can define $f^p = \sum_{l_{u,v}} -\log\left(prob(u,v)\right), l_{u,v} \in p$, where $prob(u,v)$ is the reliable probability of $l_{u,v}$, such that the most reliable path is preferred. Therefore, we call f^p the optimization function, and the logical distance computed by f^p the optimization metric, given that the path with minimal logical distance is also optimal w.r.t. the optimization requirements implied by f^p. Note that an optimization metric should not be confused with the an actual routing metric, which can be for instance bandwidth or delay.

It is worth pointing out that logical distances are **not** necessarily simple real numbers. A logical distance is a real number only if the optimization function f^p can be given by a close-form equation, as for the cases of the bandwidth-inversion shortest path ($\sum_{l_{u,v}} \frac{1}{B(u,v)}$), the most-reliable path ($\sum_{l_{u,v}} -\log\left(prob(u,v)\right)$), or the aggregated metric used in the Interior Gateway Routing Protocol (IGRP) [2]: $f^p = L + \frac{k}{C}$, where k is a positive constant, L and D are the path length and capacity, respectively. Otherwise, a logical distance can be a tuple consisting of the multiple routing metrics being considered. For example, for widest shortest-path (WSP) routing, a path with the shortest distance (e.g., hops) is preferred, and if multiple such shortest paths exist, the one with the maximal bandwidth is preferred. For WSP, f^p is defined by $(\sum D_{u,v}, min(B_{u,v}))$, in which $D_{u,v}$ and $B_{u,v}$ are the distance and bandwidth of each comprising link $l_{u,v}$ along the path. As the result, to compare the precedence between paths, only bandwidth or distance is not enough. Tuple $<D, B>$ has to be used as the logical distance for each path (D and B are distance and bandwidth, respectively), and path p with logical distance $<D1, B1>$ is better than path q with $<D2, B2>$, only if

$$D1 < D2 \vee (D1 = D2 \wedge B1 > B2) \qquad (1)$$

Similarly, in least-cost shortest-path routing (LCSP), tuple $< D, C >$ is used as the logical distance for each path, in with D is the distance and C is the cost. Function f^p is defined by $(\sum D_{u,v}, \sum C_{u,v})$ over all comprising link $l_{u,v}$ of a path. Path p with $<D1, C1>$ is preferred over path q with $<D2, C2>$, only if

$$D1 < D2 \vee (D1 = D2 \wedge C1 < C2) \qquad (2)$$

By extrapolating the real-number metrics used in conventional best-effort routing to logical distances, it is handy to compute optimal paths in the context of QoS routing, provided that a total order properly exists amongst the logical distances defined by the path function f^p being used, as we further discuss in the next section.

[1] Monotone means that the logical distance of a path cannot decrease when the path is extended, and isotone means that the order $\preceq$ between two paths must be preserved when they are prefixed or suffixed by a common third path. Readers can refer to [13] for details.

[2] A minimal routing metric means that its path measurement is determined by the minimal value of the corresponding metric of all links in the path; while an additive or multiplicative routing metric means that its path measurement equals to the sum or product of the corresponding metric of all links along the path. We do not consider multiplicative metrics because they can be translated into additive metrics by using logarithm.

3 The Multi-Constrained Path Optimization Routing Protocol (MPOR)

MPOR attempts to find the optimal path for the optimization metric defined by an optimization function f^p, and this path must also be feasible w.r.t. the given constraints, if there is any.

To achieve the optimal routing defined by f^p, MPOR uses distance-vector routing to distributively compute the shortest path w.r.t. the optimization metric calculated by f^p. However, as we discussed in Sec. 1, the shortest path computed based only on a single aggregated metric may not meet the multiple constraints being considered. Therefore, MPOR maintains an x-optimal path set for each destination. A generalized loop-free condition *o-LFC* is introduced, which allows MPOR to choose any neighbor satisfying it as the successor for a destination without causing routing loops. A path weight propagation and deduction process is used to keep track of the raw weight w^p of each path in the construction of the x-optimal path set, which enables MPOR to verify whether feasible paths can be found when a routing request arrives. Lastly, we show that the convergence of MPOR can be ensured if the optimization function f^p being used is both $monotone$ and $isotone$.

We show through extensive simulations that satisfying routing success ratios can be achieved with a small value of x (e.g., $x = 5$), and the feasible paths being found also have high probabilities to be optimal or near-optimal w.r.t. the given f^p.

3.1 Loop-freedom in MPOR

When computing the first x shortest paths for a destination, a node may use multiple neighbors as successors to reach this destination. Hence care must be taken to ensure that routing loops are not formed because nodes choose their successors distributively. Theorem 3.1 introduces the ordered loop-free condition (*o-LFC*), which is a generalized loop-free condition derived from the source node condition (SNC) in [4], for computing loop-free x-optimal path set in MPOR.

Let LD^i_j denote the logical distance from node i to destination j as known by node i. LD^i_{jk} denotes the logical distance LD^k_j from node k, which is a neighbor of node i, to destination j, as reported to node i by node k. FLD^i_j denotes the *feasible logical distance (FLD)* of node i for destination j, which is an estimate of the minimal logical distance maintained for destination j by node i.

Theorem 3.1. *(*o-LFC*)* If a total order relation $\preceq$, and the associated strict order $\prec$, can be properly established on the logical distances defined by a given f^p, then a node i is free to choose any neighboring node k from its neighbor set N^i, as the successor to reach destination j without causing routing loops, provided that the following condition is satisfied:

$$LD^i_{jk} \prec FLD^i_j,\ k \in N^i \tag{3}$$

Proof. For any neighbor k satisfying (3), $FLD^k_j \prec FLD^i_j$. Therefore, given that all estimates are properly updated, loop-freedom is obtained, because the FLDs of all the nodes along a path to destination j are strictly ordered in a decreasing manner. Once a total order can be properly established on the logical distances calculated by f^p, the formal proof of *o-LFC* is similar to that of SNC (see [4], Theorem 1, pp. 132), which is omitted here due to space limitations. □

A set of logical distances $\mathcal{D}$ is total-ordered w.r.t. $\preceq$ if the following four properties are satisfied: (1) $\preceq$ is reflexive, i.e., $a \preceq a, \forall a \in \mathcal{D}$; (2) $\preceq$ is anti-symmetric, i.e., if $a \preceq b$ and $b \preceq a$, then $a = b$; (3) $\preceq$ is transitive, i.e., if $a \preceq b$ and $b \preceq c$, then $a \preceq c$; and (4) for $\forall a, b \in \mathcal{D}$, either $a \preceq b$ or $b \preceq a$. We use $a \prec b$ to denote that $a \preceq b$ while $a \neq b$.

For example for WSP, if $LD(p)$ and $LD(q)$ are the logical distances of two paths p and q, respectively, then $LD(p) \prec LD(q)$ only if $D^p < D^q \vee (D^p = D^q \wedge B^p > B^q)$; for LCSP, $LD(p) \prec LD(q)$ only if $D^p < D^q \vee (D^p = D^q \wedge C^p < C^q)$.

When only a single additive metric is considered, $\preceq$ and $\prec$ reduce to the normal $\leq$ and $<$; and *o-LFC* reduces to the source node condition (SNC) in [4], which considers path distances in positive real numbers only, on which the total order $\leq$ and $<$ are always well defined.

3.2 Operation principles of MPOR

A node i that runs MPOR maintains a routing entry for each destination j, which includes feasible logical distance FLD^i_j, current logical distance LD^i_j and S^i_j – the successor set chosen for j. Node i maintains a neighbor table that records the logical distance LD^i_{jk} reported by each node k in its neighbor set N^i for each destination j; and a link table that reflects the link state $w(i,k)$ for each adjacent link $l_{i,k}, k \in N^i$.

Because every node must run a separate copy of MPOR and runs MPOR for each destination, we focus on node i running MPOR to compute the x-optimal path set for a destination j. It should be pointed out that i may receive and record up to x (the shortest, the second shortest and up to the xth shortest) values of LD^i_{jk} from each neighbor k; node i also reports to its neighbors the first x shortest logical distances from itself to destination j, of which the minimal value is also used as the feasible logical distance FLD^i_j of node i. For the destination j, we have $LD^j_j = \bar{0}$, $FLD^j_j = \bar{0}$, and $LD^k_{jj} = \bar{0}, \forall k \in N^j$, where $\bar{0}$ is the *zero* as defined by f^p. When a node is powered up, FLD is set

to $\bar{\infty}$, the *infinity* as defined by f^p, and all the other entries are set to empty. We also assume that node i knows the state of each outgoing link $w(i,k), k \in N^i$. When node i receives an input event at time t, it behaves in one of three possible ways:

1. Node i is idle and all estimates are left unchanged
2. Node i receives LD_j^k from neighbor k, updates the estimates LD_{jk}^i and leaves all other estimates unchanged
3. Node i updates its successors set $S_j^i(t)$ for destination j based on Eq. (3), i.e.,

$$S_j^i(t) = \{k | LD_{jk}^i(t) \prec FLD_j^i(t),\ k \in N^i\} \quad (4)$$

and updates the feasible logical distance

$$FLD_j^i(t) = min\left(LD_{jk}^i(t) \oplus ld(i,k)(t)\right) \quad (5)$$

for all LD_j^k reported by each neighbor k, and over all neighbors in N^i. $ld(i,k)$ is the logical distance of the adjacent link $l_{i,k}$, and $\oplus$ is the binary operator defined by f^p, which is used to combine two paths (or links) and compute the logical distance of the resulting composite path. Node i also re-computes the first x shortest LD_j^i maintained for j, and sends neighbors updates if any change occurs; otherwise, it leaves all other estimates unchanged. □

As with the distributed Bellman-Ford (DBF) algorithm, MPOR uses distance vectors to communicate logical distances only amongst neighboring nodes, and therefore avoids expensive routing overhead caused by disseminating link-state information throughout the network. However, major differences exist between MPOR and DBF. MPOR can optimize the routing as defined by a given optimization function f^p, as well as the simple metrics used in conventional best-effort routing. MPOR maintains the first x shortest paths for each destination, and therefore is also able to support multi-constrained path selection; Lastly, MPOR is a loop-free multipath QoS routing protocol due to the use of *o-LFC* (Theorem 3.1), which allows nodes to choose their successors independently without causing loops.

3.3 Handling network dynamics

As long as a node finds neighbors that satisfy o-LFC, it can update its routing table without having to coordinate with its neighbors (Theorem 3.1).

When node i is unable to find neighbors satisfying o-LFC for destination j, must increase FLD_j^i in a way that no loops can be created. MPOR uses a scheme similar to that used by DUAL [4] to achieve instantaneous loop freedom. More specifically, diffusing computations are used to coordinate upstream nodes in the path for destination j, when node i detects the increase of logical distance and needs to raise its feasible logical distance for j, before another set of feasible successors can be safely obtained. Due to space limitations, we omit the details here and readers can refer to [4] for more details.

3.4 Deduction of path weight w^p

For a path p in the x-optimal path set computed for destination j, besides the logical distance, its raw path weight w^p must also be maintained, because we need w^p to verify whether p can be a feasible path when a routing request arrives. To achieve this, MPOR uses a path weight propagation and deduction process to keep track of w^p, which exploits the same routing messages used for the exchange of logical distances, and as such incurs no more communication overhead.

More specifically, in MPOR, the distance vector reporting a path p for j by node k is a tuple of $\{j, LD_j^k, pw_j^k\}$, in which LD_j^k and $pw_j^k = \{pw_j^k(1), ...pw_j^k(m)...pw_j^k(C)\}$ are the logical distance and the associated path weight for p, respectively, and C is the number of constraints. Provided that $pw_j^j(m) = 0$ for an additive metric and $pw_j^j(m) = \infty$ for a minimal metric, then after receiving the distance vector from neighbor k via adjacent link $l_{i,k}$, whose link weight is $w(i,k) = \{w_1, ...w_m, ...w_C\}$, node i can compute the corresponding path weight for j by

$$pw_j^i(m) = \begin{cases} pw_j^k(m) + w_m & \text{for additive metrics} \\ min\left(pw_j^k(m), w_m\right) & \text{for minimal metrics} \end{cases} \quad (6)$$

where $m = 1, 2...C$.

3.5 Optimization function f^p and convergence to optimal paths

We argue that the choice of f^p is policy-oriented or application-oriented in practice, because there does not exist an absolutely better or best routing optimization metric for a given network. Therefore, the strategy adopted by MPOR is to specify the properties that an optimization function must have, instead of specifying a specific f^p. The advantage of this approach is that, diversified routing optimization policies can be defined and implemented as long as f^p is properly selected, and the convergence of MPOR is also ensured simultaneously. This is formally formulated in the following theorem.

Theorem 3.2. If the f^p being used is both **monotone and isotone**, within finite time after the last link-state change occurs in the network, MPOR converges correctly, i.e., maintains the optimal path for each destination.

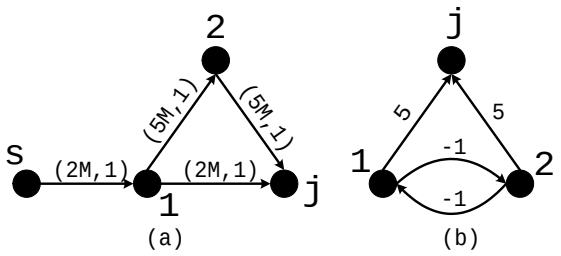

Figure 1. Counter examples – when isotonicity or monotonicity fail

Proof. According to the sufficient and necessary algebraic conditions for routing convergence [13] (Sec. 6.2, Proposition 3-5), a distance vector routing protocol always converges to optimal paths w.r.t. the optimization metric defined by a given f^p, if and only if f^p is both monotone and isotone. Because in essence MPOR exploits distance vectors and computes the optimal logical distance for each destination using Bellman-Ford iteration (Eq. (5)), the results established in [13] also hold for MPOR. □

The reason that f^p has to be monotone and isotone, for MPOR to converge correctly, can be illustrated by the examples depicted in Figure 1. We assume that nodes used in our examples run a DBF-like distance-vector protocol.

Unlike WSP, shortest-widest path (SWP) routing attempts to find the path having the maximal bandwidth, and the one with shortest distance is selected if multiple such paths are found with the same bandwidth. It is easy to verify that isotone is not true in SWP. In Figure 1 (a), links are labeled with (B,D) where B is the bandwidth and D is the distance (in hops), respectively. Node 1 chooses node 2 as the successor for destination j because path $(1, 2, j)$ offers more bandwidth (5M) than path $(1, j)$ (2M); node S has no other choice but node 1 as the successor for destination j. As a result, when each node selects its successor distributively, the path formed from S to j is $(S, 1, 2, j) =< 2M, 3 >$, which is not optimal because path $(s, 1, j) =< 2M, 2 >$ and $< 2M, 2 > \prec < 2M, 3 >$.

Figure 1 (b) is an example in which the monotone property does not hold. Assume that link $l_{1,2}$ and $l_{2,1}$ have negative cost -1, and the f^p used is simple addition $\sum lc_i$ where lc_i is the cost of every intermediate link in the path. It is easy to see that the paths for destination j at nodes 1 and 2 can oscillate and never converge. In this simple scenario, the result is obvious, because DBF cannot converge if the topology contains negative cycles [3]. More optimization functions can be found in [12, 13].

4 Complexity Analysis

At node i, the space complexity is $O(x|N^i|N + xN) = O(x|N^i|N)$, where $|N^i|$ is the number of neighbors of node i, because the main routing table and each neighbor table have $O(N)$ entries, and each entry can keep up to x routes for each destination. The computation complexity is the time taken to process distance vectors regarding a same destination, which is $O(x|N^i|)$. When only the shortest path is maintained, then the space and computation complexity reduce to $O(|N^i|N)$ and $O(|N^i|)$, respectively, which are the same as that of DBF.

5 Simulation

We implemented MPOR in the network simulator NS-2[11] and conducted extensive simulations to test its performance. Three types of topologies are used in our simulation experiments: ANSNET, *Pure-random* graph and *Waxman* graph. ANSNET is widely used by Chen and Nahrstedt [1] and other researchers to study QoS routing algorithms. In Pure-random graphs, the existence of the link between any two nodes is determined by a constant probability $\{p_r|\ 0 < p_r < 1\}$, while in Waxman graphs, p_r is defined as $p_r = \alpha e^{-\frac{d}{\beta L}}, 0 < \alpha, \beta < 1$, where d is the distance between these two nodes and L is the maximal inter-nodal distance in the topology, such that inter-nodal distance is also accounted for.

5.1 Finding feasible paths

The evaluation metrics we used in this case are success ratio (SR), existence percentage (EP) and competitive ratio (CR), which are introduced by Yuan [14] and defined as follows.

$$SR = \frac{\textit{number of routing requests being routed}}{\textit{number of total routing requests}} \quad (7)$$

$$EP = \frac{\textit{number of requests routed by exact algorithm}}{\textit{number of total routing requests}} \quad (8)$$

$$CR = \frac{\textit{number of requests routed by heuristic algorithm}}{\textit{number of requests routed by exact algorithm}} \quad (9)$$

EP actually equals the SR of an exact algorithm, and indicates how difficulty a feasible path can be found to meet the given request (a small EP means that it is hard to find a feasible path for the given constraints); CR indicates how well a heuristic algorithm can work in comparison to the exact algorithm with the same EP.

[1]The purpose of such configuration is to compare with existing simulation results of MCP algorithms, e.g. in [6].

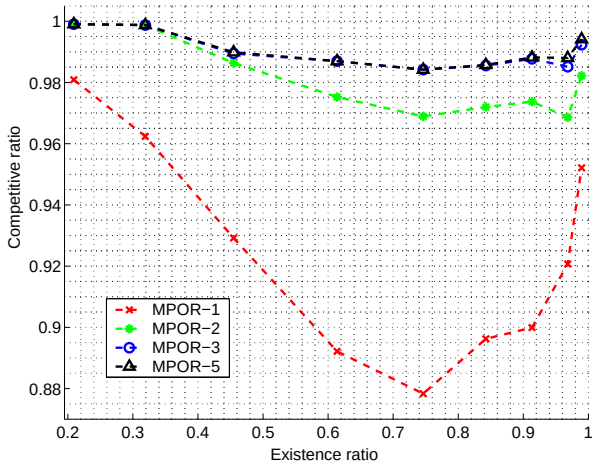

Figure 2. ANSNET (32 nodes and 54 links), two constraints, CR Vs. EP and *x = 1,2,3,5*. Randomly chosen sources and destinations

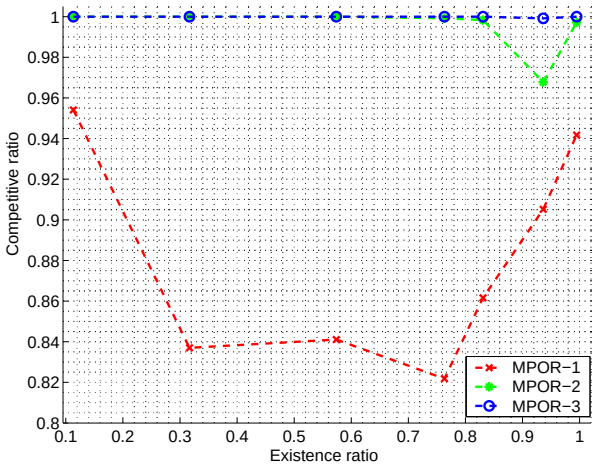

Figure 3. ANSNET (32 nodes and 54 links), two constraints, CR Vs. EP and *x = 1,2,3*. Coast-to-coast sources and destinations

For all configurations, all link weight components are uniformly distributed in $(10, 20]$, except for ANSNET in the case of two constraints, in which the first component is uniformly distributed in $(0, 50]$, while the second is uniformly distributed in $(0, 200]$[1]. All sources and destinations are randomly chosen from the networks and all results presented here are averaged over 5000 randomly generated routing requests, for different ranges of routing constraints. Routing metrics being considered are additive unless specified otherwise, because most algorithms proposed for MCP, and those being used to compare with MPOR in our simulations, only address additive constraints. The f^p used by MPOR is a linear combination that considers each link-weight component equally, which is defined by $f^p = \sum \alpha w_i, \alpha = 1/k$, if there are k constraints. It is easy to verify that this function is both monotone and isotone.

As the first step, we study how the performance of MPOR varies with the size of the optimal path set (i.e., the parameter x) maintained for each destination, using ANSNET as the topology. The results are presented in Fig. 2 and Fig. 3, respectively. Fig. 2 indicates that a 3-optimal path set for every destination is sufficient for MPOR to achieve satisfying competitive ratios (no less than 98%), for all ranges of constraints (EP ranges from 0.25 to 0.99), when sources and destinations are randomly chosen from the network. Also as shown by Fig. 3, MPOR performs even better when sources and destinations are chosen from boundary nodes, i.e., potential feasible paths traverse from coast to coast, because a 3-optimal path set configuration achieves almost 100% CRs for all ranges of constraints. We also notice that the CR of MPOR approaches to a saturation level and cannot be improved anymore even by increasing the value of x. The main reason is that the *o-LFC* condition has to be satisfied before a neighbor can be selected into the successor set, and therefore some feasible paths may be excluded due to the f^p being used. In the following, parameter $x = 5$ unless specified otherwise.

The MCP algorithms that we use to compare with MPOR are EDFS [7], KKT (Korkmaz, Krunz and Tragoudas [6]) and JSP (a variation of Jaffe's algorithm [5]: Dijkstra's shortest path algorithm w.r.t. the aggregated link-cost function $w(u,v) = [\frac{w_1(u,v)}{c_1} + \frac{w_2(u,v)}{c_2}]$), in which c_i is the constraint on routing metric w_i. As the baseline, we also implement an exact algorithm, which has exponential running time, but can give all feasible paths for a given routing request. The SR comparisons of routing with two constraints using ANSNET, Pure-random and Waxman graphs are presented in Figs. 4 and 5, respectively. The results of routing with more constraints are presented in Fig. 6.

JSP lags behind all the other algorithms in all scenarios, which further confirms that paths optimized w.r.t. a single aggregated metric may not be a good approach for constrained path selection. EDFS outperforms MPOR and other algorithms when the constraints are tight, and performs worse when the constraints are moderate and loose, for which EDFS has to iterate over more exploring sequences to achieve a better SR. Though both KKT and MPOR perform consistently and similarly well, MPOR can solve general k-constrained MCP problem, while KKT only deals with two constraints and its running time is unpredictable if constraints are tight [7]. We also observe that the SR of MPOR is insensitive to the number of constraints in all ranges of constraints and for all the topologies being simulated. Last but not least, we point out that all the algo-

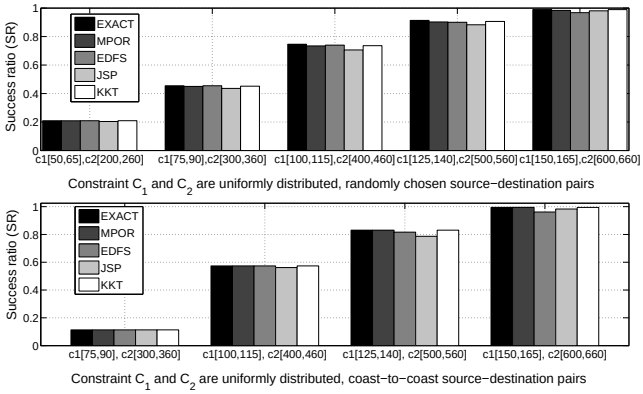

Figure 4. ANSNET, two constraints, SR comparison, *x = 5*

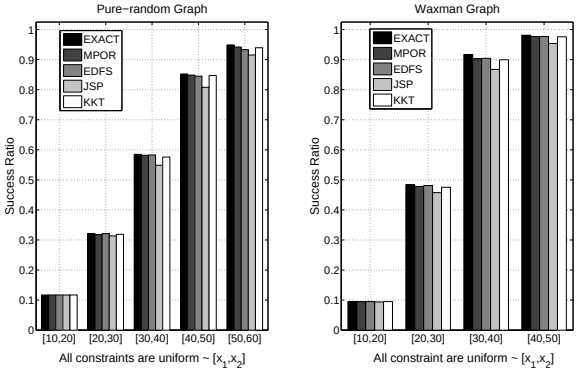

Figure 5. Pure-random (39 nodes and 75 links) and Waxman graphs (40 nodes and 95 links), two constraints, SR comparison, *x = 5*

rithms we use to compare with MPOR are centralized algorithms and require the global network state at every source node, while MPOR is a fully distributed protocol.

5.2 Finding constrained optimal paths

We investigate the 2-additive-constraint widest-path (in terms of inverted bandwidth) routing and 2-additive-constraint most-reliable path routing, which represent the scenarios in which the metrics to be optimized are minimal and multiplicative, respectively.

The challenge here is how to communicate the bandwidth $bw(u,v)$ or reliability probability $p(u,v)$ of a link $l_{u,v}$ throughout the network without using link-state broadcasting. Our approach is to define an f^p (monotone and isotone) that translates $bw(u,v)$ or $p(u,v)$ into an additive

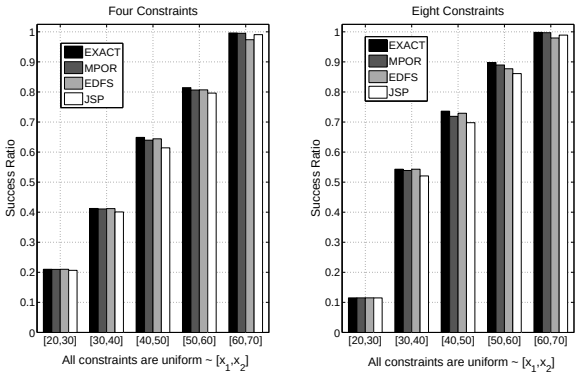

Figure 6. ANSNET, multiple constraints, SR comparison, *x = 5*

metric, and uses it as the logical distance communicated by distance vectors. More specifically, for widest-path optimization, we define f^p as

$$f^p = \sum_{l_{u,v}} \frac{1}{bw(u,v)/B_{max}}, l_{u,v} \in p \qquad (10)$$

where B_{max} is an estimate of the maximum bandwidth over all the links in the network, then bw/B_{max} can be thought of as the normalized bandwidth of link $l_{u,v}$. For most-reliable path optimization, we define f^p as

$$f^p = \sum_{l_{u,v}} -\log\left(p(u,v)\right), l_{u,v} \in p \qquad (11)$$

Figure 7 and Fig. 8 demonstrate the probability distribution functions (PDF) of how a feasible path can be found within the x-optimal path set constructed by MPOR. First, as presented by the legends in Fig. 7 and Fig. 8, both f^p (10) and f^p (11) achieve satisfying competitive ratio (CR) for the given two additive constraints. The CR for function (10) is above 94.6%, and the CR for function (11) is above 96.7%, for all ranges of existence percentage (EP), which varies between 0.05 and 0.98 (i.e., from tight constraints to loose constraints). Second, more than 70% of the feasible paths found by MPOR are also the optimal paths w.r.t. the specified f^p, and almost 95% of the feasible paths can be found if we also consider the second and third shortest paths computed by MPOR. This indicates that the feasible paths computed by MPOR have a high probability of being optimal, or near-optimal paths w.r.t. the given optimization metric. Note that in some prior work [8] MCPO is defined as selecting the shortest path from the set of feasible paths only, which therefore may not be globally optimal w.r.t. the given f^p. In this case, it is intuitive that MPOR has an even higher probability of finding the constrained optimal path.

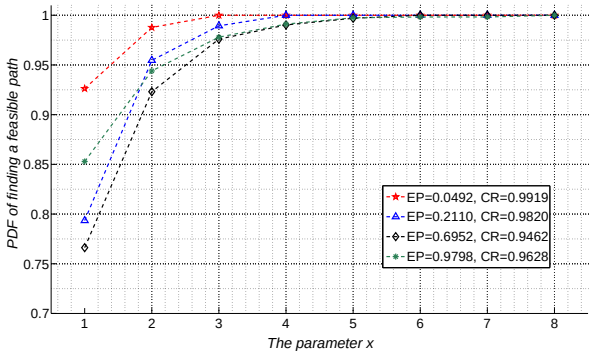

Figure 7. ANSNET, two constraints, optimize w.r.t. bandwidth

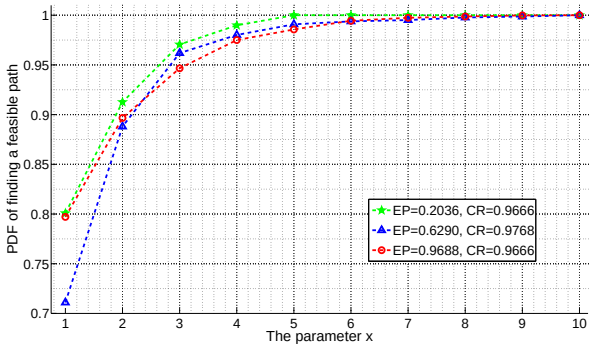

Figure 8. ANSNET, two constraints, optimize w.r.t. reliability

6 Conclusion and future work

MPOR is a fully distributed multi-constrained routing and path optimization protocol that exploits distance vector routing, and does not require global network state to be maintained at every node, or other unrealistic assumptions such as knowing the distribution of arriving constraints at each node. MPOR can optimize routing w.r.t. any logical distance computed by an optimization function f^p, provided that f^p is both $monotone$ and $isotone$.

In the future, we are interested in addressing the routing overhead incurred by diffusing computations when path distances increase or link failures occur. We are also interested in using multiple f^p for different classes of traffic flows, which is meaningful in the context of a DiffServ model, for example.

References

[1] S. Chen and K. Nahrstedt. On Finding Multi-constrained Paths. In *Proceedings of ICC'98*, pages 874–879, June, 1998.

[2] J. Doyle. *Routing TCP/IP*. Cisco Press, 1998.

[3] T. H. C. et al. *Introduction to Algorithms, Second Edition*. McGraw-Hill, 2003.

[4] J. J. Garcia-Lunes-Aceves. Loop-free Routing Using Diffusing Computations. *IEEE/ACM Trans. Netw.*, 1(1):130–141, 1993.

[5] J. M. Jaffe. Algorithms for Finding Paths with Multiple Constraints. *IEEE Networks*, 14:95–116, 1984.

[6] T. Korkmaz, M. Krunz, and S. Tragoudas. An Efficient Algorithm for Finding a Path Subject to Two Additive Constraints. In *Proceedings of the ACM SIGMETRICS*, pages 318–327, 2000.

[7] Z. Li and J. J. Garcia-Luna-Aceves. Solving the Multi-Constrained Path Selection Problem by Using Depth First Search. In *Proceedings of QSHINE'05, Orlando, Florida*, August, 2005.

[8] W. Liu, W. Lou, and Y. Fang. An Efficient Quality of Service Routing Algorithm for Delay Sensitive Applications. *Computer Networks*, 1(47):87–104, 2004.

[9] P. V. Mieghem, H. D. Neve, and F. Kuipers. Hop-by-Hop Quality of Service Routing. *Comput. Networks*, 37(3-4):407–423, 2001.

[10] H. D. Neve and P. V. Mieghem. TAMCRA: A Tunable Accuracy Multiple Constraints Routing Algorithm. *Computer Communications*, 23:667–679, 2000.

[11] NS2. the Network Simulator, http://www.isi.edu/nsnam/ns/.

[12] J. L. Sobrinho. Algebra and Algorithms for QoS Path Computation and Hop-by-Hop Routing in the Internet. *IEEE/ACM Trans. Netw.*, 10(4):541–550, 2002.

[13] J. L. Sobrinho. Network Routing with Path Vector Protocols: Theory and Application. In *Proceedings of ACM SIGCOMM*, 2003.

[14] X. Yuan. Heuristic Algorithms for Multiconstrained Quality-of-Service Routing. *IEEE/ACM Trans. Netw.*, 10(2):244–256, 2002.